

REMARKS

Claims 1-118 are currently pending in the application, claims 1-24 and 63-118 being withdrawn from consideration responsive to a previous restriction requirement, and claims 25-62 being currently examined. Claims 25 and 44 are the only independent claims currently being examined.

As a preliminary matter, should the Examiner deem the claims of the present application to be allowable, Applicant requests that Figure 35 be used as the figure for the cover sheet on the issued patent. The Examiner is invited to telephone the undersigned to discuss alternative figures should it be necessary.

The claims remain rejected as set forth in the last Office Action. In particular, claims 25-27, 29-35, 37, 39-46, 48-54, and 62 stand rejected under 35 U.S.C. §102(b) as being unpatentable over Kimpara et al. (5,310,962). Claims 28, 36, 38, 47, and 55-61 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kimpara in view of Toriumi (6,062,868). Applicant respectfully traverses these rejections, and requests reconsideration and allowance of the pending claims in view of the following arguments.

1. Overview

Applicant first takes the opportunity to characterize prosecution of the application.

A) Examiner issues a Restriction Requirement resulting in an election for which the Examiner only examines less than 40 of the original 118 claims.

B) The first Office Action of 12/28/2007 rejects all claims.

C) Applicant's first response of 3/28/2008 amended the independent claims and provided a number of arguments as to the patentability of the claims.

D) The second Office Action of 7/24/2008 addresses the previous version of the claims, does not consider several of the newly added claim features, does not specifically address Applicant's arguments, and deems a newly added claim limitation as "inherent."

The rejection is clearly deficient as will now be described in more detail.

2. Examiner's rejection covers the previous version of the claims

Claims 25-27, 29-35, 37, 39-46, 48-54, and 62 stand rejected under 35 U.S.C. §102(b) as being unpatentable over Kimpara et al. (5,310,962).

Independent claim 25 is directed toward a system for extracting and generating control

signals and recites “a first image processing element for receiving a first incoming video signal, wherein said first image processing element detects at least one image-related component within an image represented by said first incoming video signal.”

First, the present Office Action addresses a “decomposition element” and other features which are no longer recited in the claim. The Examiner’s comments with regard to such elements are therefore irrelevant to the instant claim.

Second, the rejection to claim 25 does not identify where any of the above-underlined claim elements are found in Kimpara. The rejection is clearly deficient.

3. Examiner fails to address Applicant’s arguments

Concerning claim 25, Applicant’s last response provided comments distinguishing the MIDI tempo data of Kimpara and the claimed “MIDI continuous controller message” and “MIDI note message.”

The Examiner is surely aware of MPEP § 707.07(f), which provides:

“In order to provide a complete application file history and to enhance the clarity of the prosecution history record, an examiner must provide clear explanations of all actions taken by the examiner during prosecution of an application.

Where the requirements are traversed, or suspension thereof requested, the examiner should make proper reference thereto in his or her action on the amendment.

Where the applicant traverses any rejection, the examiner should, if he or she repeats the rejection, take note of the applicant's argument and answer the substance of it.” (emphasis added).

The Examiner’s failure to address Applicant’s comments with regard to the MIDI tempo data of Kimpara is a clear violation of MPEP § 707.07(f). Applicant requests that the Examiner address Applicant’s comments in accordance with the identified MPEP requirements.

4. Inherency has not been established

Claim 25 was amended to recite “at least one of a MIDI continuous controller message and a MIDI note message, the outgoing control signal being generated responsive to content in the image represented by said first incoming video signal.” In the present Office Action, the Examiner dismisses this element as being “inherent” without providing anything to support the notion that the

claimed MIDI continuous controller message and MIDI note message are an inherent characteristic of something in Kimpara. The rejection is deficient in a number of ways.

First, the Examiner fails to identify what in Kimpara inherently discloses the claimed “MIDI continuous controller message” and the “MIDI note message.” How does an entire reference inherently disclose a claim feature?

Second, the Examiner fails to provide any reasoning to support the determination that the claimed MIDI features, which are allegedly inherent characteristics, necessarily flow from the teachings of Kimpara. MPEP 2112 provides: “In relying upon the theory of inherency, the examiner must provide a basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic necessarily flows from the teachings of the applied prior art.” *Ex parte Levy*, 17 USPQ2d 1461, 1464 (Bd. Pat. App. & Inter. 1990) (emphasis in original). Here, Examiner Fletcher provided absolutely no discussion relating to his position of inherency.

5. MIDI tempo data differs from MIDI continuous controller and MIDI note messages

The Examiner does not explicitly state that the MIDI tempo data of Kimpara is being used to teach the claimed “at least one of a MIDI continuous controller message and a MIDI note message.” The Examiner’s comments on this point are vague and simply refer to the output of the Kimpara patent. In the event this is indeed the Examiner’s position, Applicant will now demonstrate that the claimed “MIDI continuous controller message” and “MIDI note message” are distinguishable from MIDI tempo data.

Attached Appendix A is a summary of MIDI message’s provided by the MIDI Manufactures Association and is available at (<http://www.midi.org/about-midi/table1.shtml>). Table 1 shows, *inter alia*, the MIDI note message’s of Note Off event (1000nnn) and Note On event (1001nnn). Table 1 of Appendix A also shows a Timing Clock message (11111000). The Timing Clock message is sometimes referred to as MIDI tempo data. See, for example, Appendix B, which is an explanation of MIDI timing concepts and is available at (<http://web.physik.rwth-aachen.de/~harm/aixphysik/sound/midi/pages/miditmcn.html>). Lastly, Appendix C is a list of MIDI controller commands. This list is available at (<http://improv.sapp.org/doc/class/MidiOutput/controllers/controllers.html>).

The foregoing establishes that the MIDI tempo data of Kimpara (shown as a Timing Clock message in Appendix A) is entirely different than a MIDI continuous controller message (Appendix C) or a MIDI note message (Appendix A). Consequently, the MIDI tempo data

output of Kimpara cannot therefore teach or suggest “at least one of a MIDI continuous controller message and a MIDI note message,” as required by claim 25.

In view of the foregoing, Kimpara fails to teach or suggest the identified features recited in claim 25, and therefore this claim is believed to be patentable. Independent claim 44 includes language similar to that of claim 25, and thus, is believed to be patentable for reasons similar to those discussed with regard to claim 25. The rejected dependent claims are believed to be patentable at least by virtue of their dependence on their respective independent claims.

6. Examiner failed to respond to claim 44 arguments

Applicant’s last response provided comments relating to claim 44. However, the Examiner failed to respond or even acknowledge such comments in a manner required by MPEP § 707.07(f) (as set forth above). Accordingly, Applicant again submits that claim 44 recites “the outgoing control signal being generated responsive to the detected posture of the human hand.” Applicant has thoroughly reviewed Kimpara and has found no such teaching or suggestion relating to this element. Accordingly, claim 44 is believed patentable for this additional reason.

7. Rejection under 35 U.S.C. § 103(a)

Claims 28, 36, 38, 47, and 55-61 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kimpara in view of Toriumi.

Applicant has demonstrated above that Kimpara does not teach or suggest various features recited in claims 25 and 44. Applicant further submits that Toriumi does not supply any of the deficiencies of Kimpara. Therefore, for the reasons presented above, even if one skilled in the art were to combine the teachings of Kimpara and Toriumi in the manner asserted, the rejected dependent claims would be patentable at least by virtue of their respective dependencies upon patentable independent claims 25 and 44.

CONCLUSION

In view of the above, Applicant submits that the currently pending claims are in condition for allowance. However, should there remain any outstanding issues, it is respectfully requested that the Examiner telephone the undersigned so that such issues may be resolved as expeditiously as possible.

The Commissioner is hereby authorized to charge any additional fees which may be required in this application to deposit account No. 06-1135.

Address all correspondence to:
FITCH, EVEN, TABIN & FLANNERY
120 So. LaSalle Street, Ste. 1600
Chicago, IL 60603

Direct telephone inquiries to:
Jeffrey J. Lotspeich
(858) 587-7652
San Diego, California Office of
FITCH, EVEN, TABIN & FLANNERY

Respectfully submitted,

FITCH, EVEN, TABIN & FLANNERY

/Jeffrey J. Lotspeich/

Jeffrey J. Lotspeich
Registration No. 45,737
Attorney for Applicant

Date: October 24, 2008

Table 1 of 4 - Summary of MIDI Messages

The following table lists many of the major MIDI messages in numerical (binary) order. This table is intended as an overview of MIDI, and is by no means complete.

See also:

- [Table 2](#) - Expanded Messages List (Status Bytes)
- [Table 3](#) - Summary of Control Change Messages (Data Bytes)
- [Table 4](#) - Defined Universal System Exclusive Messages

Additional messages are listed in the printed documentation available from the MMA.

WARNING! Details about implementing these messages can dramatically impact compatibility with other products. We strongly recommend consulting the official [Complete MIDI 1.0 Detailed Specification](#) for additional information.

Table 1: MIDI 1.0 Specification Message Summary
Updated 1995 By the MIDI Manufacturers Association

Status D7-----D0	Data Byte(s) D7-----D0	Description

Channel Voice Messages [nnnn = 0-15 (MIDI Channel Number 1-16)]		

1000nnnn	0kkkkkkk 0vvvvvvv	Note Off event. This message is sent when a note is released (ended). (kkkkkkk) is the key (note) number. (vvvvvvv) is the velocity.
1001nnnn	0kkkkkkk 0vvvvvvv	Note On event. This message is sent when a note is depressed (start). (kkkkkkk) is the key (note) number. (vvvvvvv) is the velocity.
1010nnnn	0kkkkkkk 0vvvvvvv	Polyphonic Key Pressure (Aftertouch). This message is most often sent by pressing down on the key after it "bottoms out". (kkkkkkk) is the key (note) number. (vvvvvvv) is the pressure value.
1011nnnn	0ccccccc 0vvvvvvv	Control Change. This message is sent when a controller value changes. Controllers include devices such as pedals and levers. Controller numbers 120-127 are reserved as "Channel Mode Messages" (below). (ccccccc) is the controller number (0-119). (vvvvvvv) is the controller value (0-127).
1100nnnn	0pppppppp	Program Change. This message sent when the patch number changes.

(ppppppp) is the new program number

APPENDIX A

1101nnnn 0vvvvvvv Channel Pressure (After-touch).
This message is most often sent by pressing down on the key after it "bottoms out". This message is different from polyphonic after-touch. Use this message to send the single greatest pressure value (of all the current depressed keys). (vvvvvvv) is the pressure value.

1110nnnn 01111111
 0mmmmmmmm Pitch Wheel Change.
This message is sent to indicate a change in the pitch wheel. The pitch wheel is measured by a fourteen bit value. Center (no pitch change) is 2000H. Sensitivity is a function of the transmitter.
(111111) are the least significant 7 bits.
(mmmmmm) are the most significant 7 bits.

Channel Mode Messages (See also Control Change, above)

1011nnnn 0ccccccc
 0vvvvvvv Channel Mode Messages.
This the same code as the Control Change (above), but implements Mode control and special message by using reserved controller numbers 120-127.
The commands are:

All Sound Off.
When All Sound Off is received
all oscillators will turn off, and
their volume envelopes are set to
zero as soon as possible.
c = 120, v = 0: All Sound Off

Reset All Controllers.
When Reset All Controllers is received,
all controller values are reset to their
default values. (See specific Recommended
Practices for defaults).
c = 121, v = x: Value must only be zero
 unless otherwise allowed
 in a specific Recommended
 Practice.

Local Control.
When Local Control is Off, all devices
on a given channel will respond only to
data received over MIDI. Played data, etc.
will be ignored. Local Control On
restores the functions of the normal
controllers.
c = 122, v = 0: Local Control Off
c = 122, v = 127: Local Control On

All Notes Off.
When an All Notes Off is received,
all oscillators will turn off.
c = 123, v = 0: All Notes Off
(See text for description of actual
mode commands.)
c = 124, v = 0: Omni Mode Off
c = 125, v = 0: Omni Mode On
c = 126, v = M: Mono Mode On (Poly Off)

where M is the number of channels
 (Omni Off) or 0 (Omni On)
 c = 127, v = 0: Poly Mode On (Mono Off)
 (Note: These four messages also cause
 All Notes Off)

APPENDIX A

----- System Common Messages -----

11110000	0iiiiiii 0ddddd 0ddddd 11110111	System Exclusive. This message makes up for all that MIDI doesn't support. (iiiiiii) is usually a seven-bit Manufacturer's I.D. code. If the synthesizer recognizes the I.D. code as its own, it will listen to the rest of the message (ddddd). Otherwise, the message will be ignored. System Exclusive is used to send bulk dumps such as patch parameters and other non-spec data. (Note: Real-Time messages ONLY may be interleaved with a System Exclusive.) This message also is used for extensions called Universal Exclusive Messages.
11110001		Undefined. (Reserved)
11110010	01111111 0mmmmmm	Song Position Pointer. This is an internal 14 bit register that holds the number of MIDI beats (1 beat= six MIDI clocks) since the start of the song. l is the LSB, m the MSB.
11110011	0sssssss	Song Select. The Song Select specifies which sequence or song is to be played.
11110100		Undefined. (Reserved)
11110101		Undefined. (Reserved)
11110110		Tune Request. Upon receiving a Tune Request, all analog synthesizers should tune their oscillators.
11110111		End of Exclusive. Used to terminate a System Exclusive dump (see above).

----- System Real-Time Messages -----

11111000	Timing Clock. Sent 24 times per quarter note when synchronization is required (see text).
11111001	Undefined. (Reserved)
11111010	Start. Start the current sequence playing. (This message will be followed with Timing Clocks).
11111011	Continue. Continue at the point the sequence was Stopped.

APPENDIX A

11111100	Stop. Stop the current sequence.
11111101	Undefined. (Reserved)
11111110	Active Sensing. Use of this message is optional. When initially sent, the receiver will expect to receive another Active Sensing message each 300ms (max), or it will be assume that the connection has been terminated. At termination, the receiver will turn off all voices and return to normal (non-active sensing) operation.
11111111	Reset. Reset all receivers in the system to power-up status. This should be used sparingly, preferably under manual control. In particular, it should not be sent on power-up.

MIDI Timing Concepts



Explanation of Absolute and Relative Time
MIDI Timing Clock

Different types of Time Code

- SMPTE Time Code
- MIDI Time Code

Examples of MIDI Time Code in a Sequence
Studio design using SMPTE/MIDI Time Code
Related Terms

The first concept that needs to be understood is the difference between **absolute** and **relative time**. MIDI Time Code and SMPTE Time Code are representations of absolute time in that they follow hours, minutes and seconds just like your watch. Absolute time is always the same and you cannot speed it up or slow it down. Relative time is a reference to a musical piece that has an inner tempo. A composition may take three minutes to perform at a tempo of 80 bpm (beats per minute), but would take only a minute and a half if the tempo was increased to 160 bpm. An advanced MIDI sequencer is able to work with both absolute time and relative time and make adjustments when there are changes in the relative time of a composition.

In the chart below the upper line represents absolute time and includes an example of 17 seconds of time. Both MIDI Time Code and SMPTE Time Code may be used to represent absolute time which is fixed and may not be moved. Following the absolute time line are examples of repeated quarter notes at 3 different tempos (relative time). Notice that 2 measures have passed at 4 seconds with a tempo of 120 bps, but it would take 8 seconds for the same two measures at a tempo of 60 bpm and 16 seconds for the same two measures at a tempo of 30 bpm.

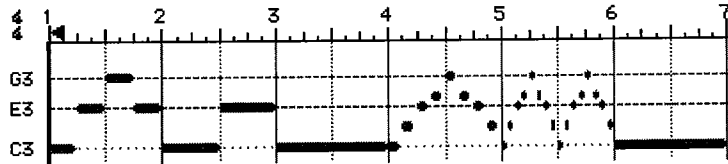
Comparing Absolute Time to Relative Time

MTC / SMPTE	Seconds	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Measure	1			2		3		4		5		6		7		8		9
120 BPM		♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪	♪
Measure	1				2				3				4				5	
60 BPM		♪		♪		♪		♪		♪		♪		♪		♪		♪
Measure	1								2									3
30 BPM		♪			♪			♪				♪			♪			♪

☐ [return to the menu](#)

MIDI Timing Clock (MIDI Sync) is a status byte (F8) that is sent 24 times per quarter note for note resolution. Advanced sequencers will also subdivide each MIDI clock twenty times for a resolution of 480 times per quarter note. Standard - 24 PPQ. Professional - 480 PPQ. If the tempo changes, the speed of the MIDI Timing Clock will pass at a faster rate, but the number of status bytes (F8) per quarter note will stay the same.

APPENDIX B



An F8 status byte is a system real time message that is used in a MIDI sequence to connect all the different

- 1. 1. 0 C3
- 1. 2. 0 E3
- 1. 3. 0 G3
- 1. 4. 0 E3
- 2. 1. 0 C3
- 2. 3. 0 E3
- 3. 1. 0 C3
- 4. 1. 0 C3
- 4. 1.240 D3
- 4. 2. 0 E3
- 4. 2.240 F3
- 4. 3. 0 G3
- 4. 3.240 F3
- 4. 4. 0 E3
- 4. 4.240 D3
- 5. 1. 0 C3
- 5. 1.120 D3
- 5. 1.240 E3
- 5. 1.360 F3
- 5. 2. 0 G3
- 5. 2.120 F3
- 5. 2.240 E3
- 5. 2.360 D3
- 5. 3. 0 C3
- 5. 3.120 D3
- 5. 3.240 E3
- 5. 3.360 F3
- 5. 4. 0 G3
- 5. 4.120 F3
- 5. 4.240 E3
- 5. 4.360 D3
- 6. 1. 0 C3

consisting of quarter notes in the 1st measure, half notes in the 2nd measure, a whole note in the 3rd measure,

followed by eighth-notes and sixteenth notes in the 4th

and 5th measure, and finally a whole note in the last

measure.

To the left is a graph depicting the same information in an

event listing. Notice that the first number represents the

measure, followed by the beat and finally the F8 timing

clock resolution. In the first three measures each note

starts exactly on the beat which corresponds to a timing

clock reference of 0. Notice at the 4th measure that the

timing clock has changed to values of 240. Remember that

480 times is the total for each quarter note and the 240

value is an eighth note. In measure 5, the value changes to

120 which now represents the sixteenth note.

If the tempo of this sequence was changed to a different

number, the numbers and values on these two charts

would not change. What would change is the location of

the notes in respect to absolute time. The value of the new

tempo would map out a new chart for placement along the

MIDI Time Code.

Now we will look at what MIDI Time Code and SMPTE

Time Code have in common (absolute time) and how it is

used in conjunction with MIDI Timing Clock (relative

time).

[return to the menu](#)

A **Time Code** is an electrical or digital signal that gives a common timing reference for syncing MIDI devices with other electronic devices. The evolution of timing references has changed dramatically since the beginning of MIDI, from the use of click track, PPQ Clocks, FSK Clocks, to the more sophisticated use of SMPTE Time Code and MIDI Time Code.

SMPTE Time Code (Society of Motion Picture and Television Engineers) Was originally developed by NASA to sync computers together. SMPTE is a digitally encoded labeling system that strips a tape with an exact timing reference. It is a 1200 Hz modulated square wave that uses a biphasic modulation to encode the signal on a tape. SMPTE is used to sync video with music, audio and sound effects. It may be used with a multi-track tape recorder and a synchronizer to sync a MIDI sequence with the tape or to sync two multi-track tape recorders. It is important to remember that a SMPTE Time Code signal is made up of digital words that contain 80 bits. These bits are used to represent the type of SMPTE signal, the actual location

which includes hours, minutes, seconds, and frames, as well as user bits for dates and other numbers. This digital number is converted to an audio tone so that it may be recorded on to an analog tape.

There are two different ways to record a SMPTE Time Code on to a tape. **LTC Time Code (Longitudinal Time Code)** is SMPTE Time Code encoded on one of the audio tracks or in-between the audio tracks of a video tape. This type of time code needs to be running in order to be read. A window burn is used on a working copy of a tape with LTC in order to address the code in slow or paused position. **VITC Time Code (Vertical Interval Time Code)** is SMPTE Time Code encoded on the Video signal in the Interval between frames. This allows for the use of SMPTE at very slow speeds without the need of a window burn or the loss of an audio track.

There are **four different SMPTE formats** in the standard. Each format refers to the number of frames per second. In film there are **24** frames which corresponds to the number of pictures per second. In European film and video the number of frames per second is **25**. In the United States the original black and white TV programs ran at exactly **30** frames per second. This changed with the advent of color to run at **29.97** frames per second. In order to compensate for this discrepancy, there is a SMPTE format called "**drop frame**" at 30 frames per second. There are exactly 30 frames for every second, except two frames are dropped at the beginning of every minute, except at minutes 0, 10, 20, 30, 40 and 50. A total of 108 frames are dropped for each minute to adjust to the 29.97 frames per second. The standard that is used for most applications is "30 drop frame" and listed below are some examples.

01:06:12:09:06

The number to the left represents a specific moment in time. The number represents 1 hour, 6 minutes, 12 seconds, 9 frames and 6 bits. There are 80 bits for every frame.

The number to the right represents 23 hours, 59 minutes, 59 seconds, 29 frames, and 79 bits. By adding one more bit to this number would change the number to 24 hours.

23:59:59:29:79

When MIDI was invented there was no specific data for synchronizing with SMPTE Time Code. If a musician was trying to use MIDI sequences to synchronize with a video tape they needed a sync device. **Song Position Pointer** is a System Common status byte F2, that is generated every six MIDI clocks. It is used as a reference to find a location in a song by counting the Song Position Pointer total and diving by 16 (4 per quarter note in a 4/4 time signature) to find the measure. A **Tempo Map** is used by a sequencer to change tempo at certain locations in a sequence. In order to synchronize with Song Position Pointer, a **sync device** is used to decide tempo and tempo changes in a song and the amount of measures in a song, using SPP and tempo map. The sync device can then figure out the amount of SMPTE time that passes at certain "hits". This was a very complicated way of synchronizing a MIDI sequence to a video tape, so in 1987, MIDI Time Code, was added to the MIDI specification in order to have a direct link of MIDI with SMPTE Time Code.

[return to the menu](#)

MIDI Time Code is a digital conversion of the SMPTE Time Code that allows MIDI devices to lock to the

SMPTE Time Code in real time. Every frame of the SMPTE Time Code is broken down into 4 frames of MIDI Time Code with an average of 120 times a second. There are System Common Quarter - Frame messages which are used when the system is running and System Exclusive Full Messages to address any location.

MIDI Time Code Quarter Frame Messages (QFM) are used when a system is running. It takes a total of 8 QFM to address one SMPTE Frame. That means that SMPTE is updated every two frames. Each QFM starts with the F1 Status byte, followed by a Data byte where the first nibble will be 0 through 7 followed by a nibble that is the actual number.

- F1** - MIDI Time Code Quarter Frame Address - Status Byte
 n= First nibble - 0 - Frame count LS nibble

 n= First nibble - 1 - Frame count MS nibble

 n= First nibble - 2 - Seconds count LS nibble

 n= First nibble - 3 - Seconds count MS nibble

 n= First nibble - 4 - Minutes count LS nibble

 n= First nibble - 5 - Minutes count MS nibble

 n= First nibble - 6 - Hours count LS nibble

 n= First nibble - 7 - Hours count MS nibble and SMPTE Type

 x= Second nibble - Actual number of the Time Code 0 through 9

nx - Data Byte

The relationship between the SMPTE time code numbers and MIDI time code quarter frame message is listed in the example below.

01:16:25:12:24

•

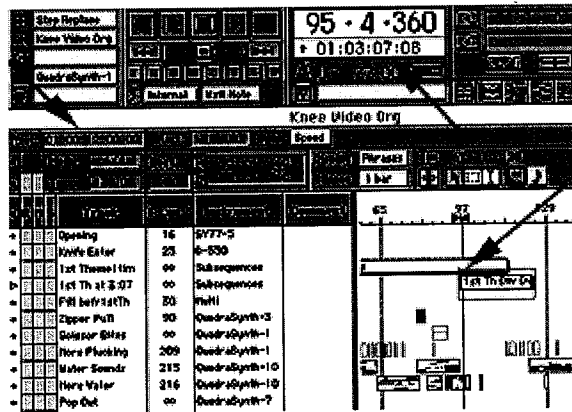
0	1	1	6	2	5	1	2
F1	F1	F1	F1	F1	F1	F1	F1
70	61	51	46	32	25	11	02

Example of MIDI Time Code in a MIDI Sequencer

APPENDIX B

In the example to the right are red arrows pointing to specific MIDI time code data in a sequencer track. The upper right arrow is pointing to a specific absolute time number of 1 hour, 3 minutes, 7 seconds and 8 frames. Below this arrow is an arrow pointing to the music that is being played at this location.

The arrow on the left is pointing to an offset number. **Offset** is a way of setting the MTC time at a certain number to begin a sequence or cue.



Here is an event listing of the notes and the arrows are pointing to an exact moment in time. In the top arrows the note D2 is played in measure 39, beat 1, with a MIDI clock resolution of 70. This note occurs at 1 hour, 1 minute, 22 seconds, 6 frames and 51 bit resolution.

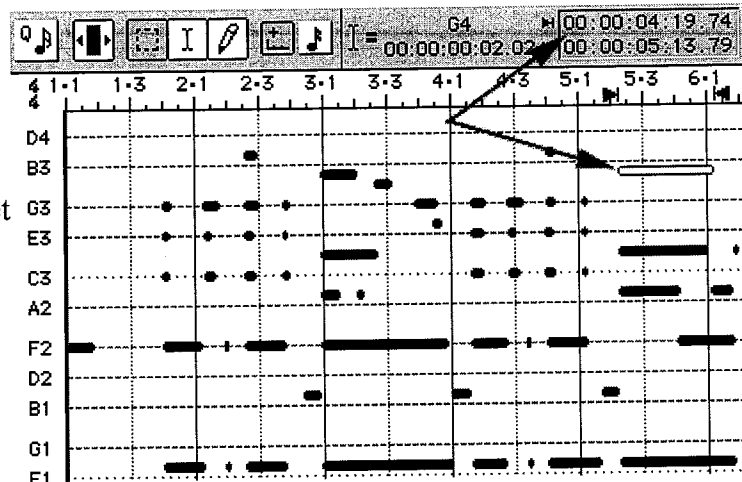
95	4	360	01:03:07:08:06	QuadraSynth-1
95	4	360	01:03:07:08:06	QuadraSynth-1
210	1	0	01:06:37:25:29	QuadraSynth-1
6	1	2	01:00:21:06.79	A2 0 • 98
32	4	280	01:01:10:26.43	D3 0 • 95
34	1	350	01:01:13:07.63	G#2 0 • 124
35	3	67	01:01:15:22.56	F#4 0 • 95
35	3	84	01:01:15:23.15	F#4 0 • 69
36	2	180	01:01:17:07.40	D2 0 • 268
39	1	70	01:01:22:06.51	D2 0 • 128
39	1	81	01:01:22:06.76	C2 0 • 78
39	3	219	01:01:23:08.50	D2 0 • 147
40	3	383	01:01:25:08.59	F#2 0 • 137
43	2	258	01:01:30:07.35	C#2 0 • 111
44	2	256	01:01:32:02.61	E#2 0 • 106
46	3	156	01:01:36:04.39	G#2 0 • 135
48	1	220	01:01:38:29.34	C2 0 • 97
48	1	222	01:01:38:29.38	F3 0 • 104
48	1	226	01:01:38:29.48	A3 0 • 117
48	1	226	01:01:38:29.48	D2 0 • 80
48	1	239	01:01:38:29.78	G3 0 • 37
51	2	447	01:01:45:05.77	B1 0 • 113
51	2	454	01:01:45:06.13	E3 0 • 115
51	2	460	01:01:45:06.27	G3 0 • 120
53	1	109	01:01:48:03.12	D2 0 • 93
53	1	115	01:01:48:03.25	A2 0 • 80
53	1	115	01:01:48:03.25	G3 0 • 93
53	1	120	01:01:48:03.37	B3 0 • 93

In the lower arrows the note C2 is played in measure 48, beat 1, with a MIDI clock resolution of 220. The time reference of this note is 1 hour, 1 minute, 38 seconds, 29 frames, and 34 bit resolution.

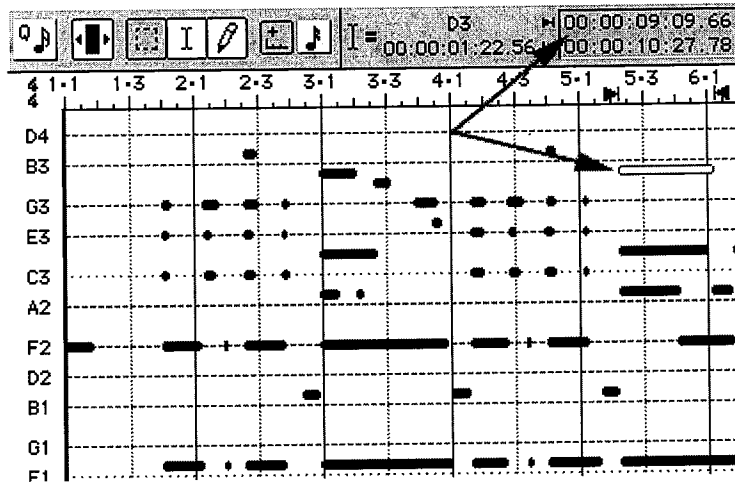
It is important to remember that if the tempo of the music changed, the time code numbers would change, but the value and placement of the notes would stay the same. The next two example clearly show the relationship of the MIDI time code with the MIDI timing clock.

APPENDIX B

There are red arrows in the diagram to the right that are locating a highlighted note in this MIDI sequence at measure 5, beat 2. The tempo for this sequence is set at **220 bpm** and the time code (upper right arrow indicates that the note will start at 4 seconds, 19 frames and 74 bits and will end at 5 seconds, 13 frames and 79 bits.



This example is the exact same sequence as above with the tempo slowed down to **110 bps**. Again the same note at measure 5, beat 2 is highlighted, but with the slower tempo the time code numbers have changed. The note will first start at 9 seconds, 9 frames and 66 bits and end at 10 seconds, 27 frames and 78 bits.



The highlighted note would start 4 seconds, 19 frames, and 72 bits into the second example with the tempo cut in half. The duration of the highlighted note in the second example would be for 1 second, 18 frames and 12 bits, while the duration of the highlighted note in the first example would only be 0 seconds, 24 frames and 5 bits. The use of MIDI Time Code in a sequencer allows the user to make changes with the tempo of a composition and all the timing information is automatically updated to follow the new tempo marking. With the use of MIDI Time Code to sync up to SMPTE Time Code, there is no need for a sync device and synchronization with Song Position Pointer

[return to the menu](#)

[Studio design using SMPTE/MIDI](#)

[Time Code](#)

[Related Terms](#)



[What is MIDI?](#)

[MIDI Connections Java Enabled](#)

[MIDI Connections Non-Java](#)

[Understanding Decimal Binary & Hexadecimal](#)

[The MIDI Language](#)

[Types of Data Transmitted through MIDI](#)

[MIDI Channels and Modes](#)

[MIDI Controllers](#)

[General MIDI](#)

[Standard MIDI Files](#)

[Using MIDI on a Web Site](#)

[Applications that use MIDI](#)

[Audio vs. MIDI Files](#)

[MIDI Timing Concepts](#)

[Author Info and Comments](#)

MIDI Continuous Controllers

A MIDI continuous controller command consists of the MIDI controller command followed by two data bytes that specify the controller number and the controller's value:

```
0xb0 | channel = MIDI continuous controller command
0 .. 127      = MIDI continuous controller number
0 .. 127      = MIDI continuous controller value
```

Controller

Number

Hex Dec Controller Name

Data Range

00	0	<u>Bank Select</u> (coarse)	0..127
01	1	<u>Modulation Wheel</u> (coarse)	0..127
02	2	<u>Breath Control</u> (coarse)	0..127
03	3	Continuous controller #3	0..127
04	4	<u>Foot Controller</u> (coarse)	0..127
05	5	<u>Portamento Time</u> (coarse)	0..127
06	6	<u>Data Entry Slider</u> (coarse)	0..127
07	7	<u>Main Volume</u> (coarse)	0..127
08	8	<u>Stereo Balance</u> (coarse)	0..127
09	9	Continuous controller #9	0..127
0A	10	<u>Pan</u> (coarse)	0=left 64=center 127=right
0B	11	<u>Expression</u> (sub-Volume) (coarse)	0..127
0C	12	<u>Effect Control 1</u> (coarse)	0..127
0D	13	<u>Effect Control 2</u> (coarse)	0..127
0E	14	Continuous controller #14	0..127
0F	15	Continuous controller #15	0..127
10	16	<u>General Purpose Slider 1</u>	0..127
11	17	<u>General Purpose Slider 2</u>	0..127
12	18	<u>General Purpose Slider 3</u>	0..127
13	19	<u>General Purpose Slider 4</u>	0..127
14	20	Continuous controller #20	0..127
15	21	Continuous controller #21	0..127
16	22	Continuous controller #22	0..127
17	23	Continuous controller #23	0..127
18	24	Continuous controller #24	0..127
19	25	Continuous controller #25	0..127
1A	26	Continuous controller #26	0..127
1B	27	Continuous controller #27	0..127
1C	28	Continuous controller #28	0..127
1D	29	Continuous controller #29	0..127
1E	30	Continuous controller #30	0..127
1F	31	Continuous controller #31	0..127
20	32	<u>Bank Select</u> (fine)	0..127 usually ignored
21	33	<u>Modulation Wheel</u> (fine)	0..127
22	34	<u>Breath Control</u> (fine)	0..127
23	35	Continuous controller #3 (fine)	0..127
24	36	<u>Foot Controller</u> (fine)	0..127
25	37	<u>Portamento Time</u> (fine)	0..127
26	38	<u>Data Entry Slider</u> (fine)	0..127
27	39	<u>Main Volume</u> (fine)	0..127 usually ignored
28	40	<u>Stereo Balance</u> (fine)	0..127
29	41	Continuous controller #9 (fine)	0..127
2A	42	<u>Pan</u> (fine)	0..127 usually ignored
2B	43	<u>Expression</u> (sub-Volume) (fine)	0..127 usually ignored
2C	44	<u>Effect Control 1</u> (fine)	0..127
2D	45	<u>Effect Control 2</u> (fine)	0..127

APPENDIX C

2E	46	Continuous controller #14 (fine)	0..127	
2F	47	Continuous controller #15 (fine)	0..127	
30	48	Continuous controller #16	0..127	
31	49	Continuous controller #17	0..127	
32	50	Continuous controller #18	0..127	
33	51	Continuous controller #19	0..127	
34	52	Continuous controller #20 (fine)	0..127	
35	53	Continuous controller #21 (fine)	0..127	
36	54	Continuous controller #22 (fine)	0..127	
37	55	Continuous controller #23 (fine)	0..127	
38	56	Continuous controller #24 (fine)	0..127	
39	57	Continuous controller #25 (fine)	0..127	
3A	58	Continuous controller #26 (fine)	0..127	
3B	59	Continuous controller #27 (fine)	0..127	
3C	60	Continuous controller #28 (fine)	0..127	
3D	61	Continuous controller #29 (fine)	0..127	
3E	62	Continuous controller #30 (fine)	0..127	
3F	63	Continuous controller #31 (fine)	0..127	
40	64	<u>Hold pedal (Sustain) on/off</u>	0..63=off	64..127=on
41	65	<u>Portamento on/off</u>	0..63=off	64..127=on
42	66	<u>Sustenuto Pedal on/off</u>	0..63=off	64..127=on
43	67	<u>Soft Pedal on/off</u>	0..63=off	64..127=on
44	68	<u>Legato Pedal on/off</u>	0..63=off	64..127=on
45	69	<u>Hold Pedal 2 on/off</u>	0..63=off	64..127=on
46	70	<u>Sound Variation</u>	0..127	
47	71	<u>Sound Timbre</u>	0..127	
48	72	<u>Sound Release Time</u>	0..127	
49	73	<u>Sound Attack Time</u>	0..127	
4A	74	<u>Sound Brightness</u>	0..127	
4B	75	<u>Sound Control 6</u>	0..127	
4C	76	<u>Sound Control 7</u>	0..127	
4D	77	<u>Sound Control 8</u>	0..127	
4E	78	<u>Sound Control 9</u>	0..127	
4F	79	<u>Sound Control 10</u>	0..127	
50	80	<u>General Purpose Button</u>	0..63=off	64..127=on
51	81	<u>General Purpose Button</u>	0..63=off	64..127=on
52	82	<u>General Purpose Button</u>	0..63=off	64..127=on
53	83	<u>General Purpose Button</u>	0..63=off	64..127=on
54	84	Undefined on/off	0..63=off	64..127=on
55	85	Undefined on/off	0..63=off	64..127=on
56	86	Undefined on/off	0..63=off	64..127=on
57	87	Undefined on/off	0..63=off	64..127=on
58	88	Undefined on/off	0..63=off	64..127=on
59	89	Undefined on/off	0..63=off	64..127=on
5A	90	Undefined on/off	0..63=off	64..127=on
5B	91	<u>Effects Level</u>	0..127	
5C	92	<u>Tremulo Level</u>	0..127	
5D	93	<u>Chorus Level</u>	0..127	
5E	94	<u>Celeste (Detune) Level</u>	0..127	
5F	95	<u>Phaser Level</u>	0..127	
60	96	<u>Data entry +1</u>	ignored	
61	97	<u>Data entry -1</u>	ignored	
62	98	<u>Non-Registered Parameter Number (coarse)</u>	0..127	
63	99	<u>Non-Registered Parameter Number (fine)</u>	0..127	
64	100	<u>Registered Parameter Number (coarse)</u>	0..127	
65	101	<u>Registered Parameter Number (fine)</u>	0..127	
66	102	Undefined	?	
67	103	Undefined	?	
68	104	Undefined	?	
69	105	Undefined	?	
6A	106	Undefined	?	
6B	107	Undefined	?	
6C	108	Undefined	?	
6D	109	Undefined	?	
6E	110	Undefined	?	

APPENDIX C

6F 111	Undefined	?
70 112	Undefined	?
71 113	Undefined	?
72 114	Undefined	?
73 115	Undefined	?
74 116	Undefined	?
75 117	Undefined	?
76 118	Undefined	?
77 119	Undefined	?
78 120	<u>All Sound Off</u>	ignored
79 121	<u>All Controllers Off</u>	ignored
7A 122	<u>Local Keyboard On/Off</u>	0..63=off 64..127=on
7B 123	<u>All Notes Off</u>	ignored
7C 124	<u>Omni Mode Off</u>	ignored
7D 125	<u>Omni Mode On</u>	ignored
7E 126	<u>Monophonic Mode On</u>	**
7F 127	<u>Polyphonic Mode On</u> (mono=off)	ignored

**Note: This equals the number of channels, or zero if the number of channels equals the number of voices in the receiver.

craig@ccrma.stanford.edu